

บทที่ 3

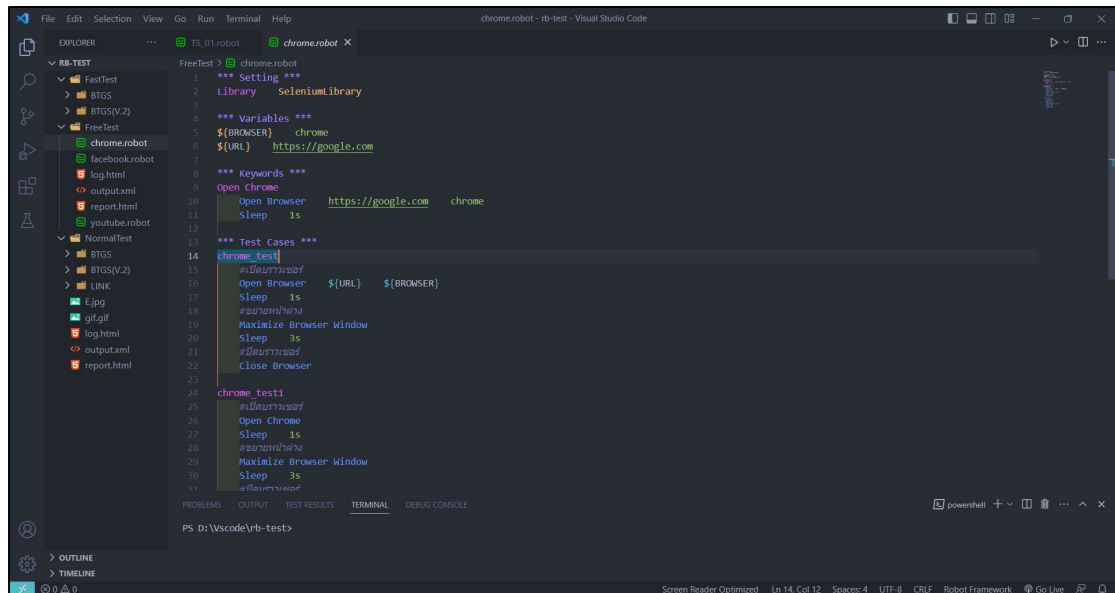
เทคโนโลยีที่ใช้ในการศึกษา

ระบบการจัดการควบคุมต้นทุนโครงการก่อสร้าง เป็นระบบที่อยู่ในกลุ่มเว็บแอปพลิเคชัน ได้นำเทคโนโลยีมาใช้ในการพัฒนาการทดสอบอัตโนมัติ ดังนี้

3.1 เทคโนโลยีที่ใช้ในการเขียน ชุดคำสั่งที่ใช้ในการทดสอบระบบ (Test Script)

3.1.1 Visual Studio Code

Visual Studio Code (VSCode) คือโปรแกรมแก้ไข และพัฒนาโค้ด (Code Editor) ที่พัฒนาโดย Microsoft ซึ่งเป็นเครื่องมือที่นักพัฒนาสามารถใช้ในการเขียน และแก้ไขโค้ด ไม่ว่าจะเป็นโค้ดภาษาโปรแกรมต่าง ๆ เช่น JavaScript, Python, Java, C++, HTML, CSS และอื่น ๆ รวมถึงการสนับสนุนสำหรับเฟรมเวิร์ค (Framework) ต่าง ๆ ที่มักถูกนำมาใช้ในการพัฒนาแอปพลิเคชันต่าง ๆ โดยในด้านการทำ Automated Testing (การทดสอบอัตโนมัติ) Visual Studio Code (VSCode) เป็นเครื่องมือที่สามารถช่วยในการพัฒนา และจัดการกับสคริปต์ที่ใช้ในการเขียน และทดสอบการทำงานของซอฟต์แวร์ โดยเฉพาะในส่วนของ Unit Test และ End-to-End Testing นอกจากนี้ยังมี Extensions หลายตัวที่สามารถช่วยในการเขียน และเรียกใช้งาน Automated Testing ซึ่งจะช่วยในการจัดการข้อมูลการทดสอบ การรันทดสอบอัตโนมัติ การสร้างรายงานผลการทดสอบ และฟีเจอร์อื่น ๆ ที่ช่วยในกระบวนการทดสอบซอฟต์แวร์อย่างมีประสิทธิภาพ และมีความน่าเชื่อถือ

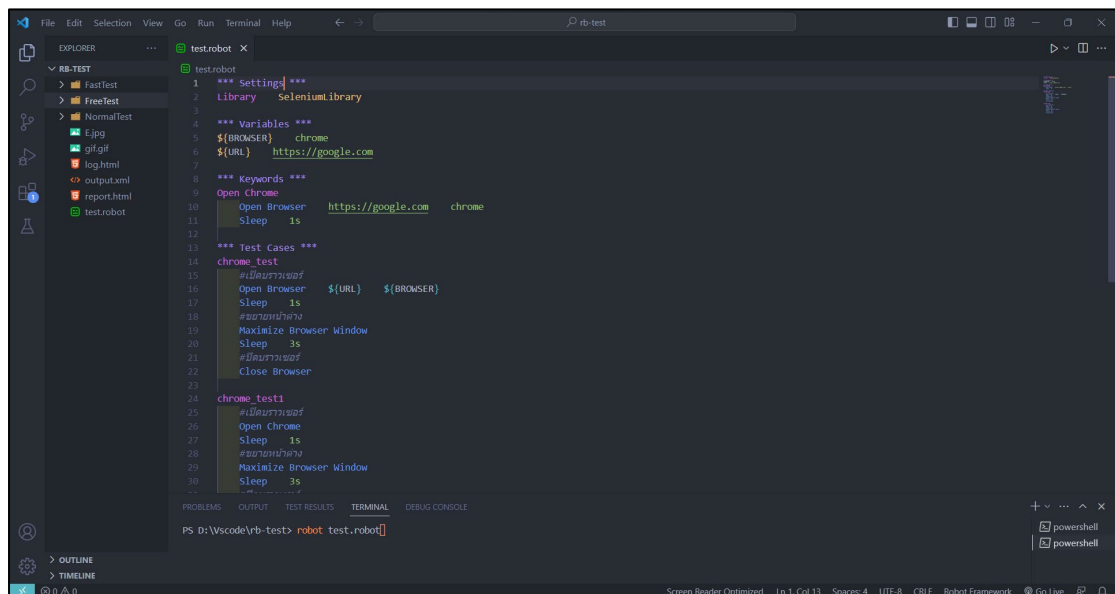


รูปที่ 3.1 ตัวอย่างหน้าการใช้งานโปรแกรม Visual Studio Code (VSCode)

3.2 เทคโนโลยีที่ใช้ในการทดสอบระบบ

3.2.1 Robot Framework

Robot Framework เป็นเครื่องมือที่ใช้ในการทดสอบอัตโนมัติ โดยใช้คำสั่งในการทำงาน และคำสั่งที่ใช้พัฒนามาจากภาษา Python ซึ่งถูกพัฒนาขึ้นเพื่อช่วยในการทดสอบและตรวจสอบความถูกต้องของซอฟต์แวร์ต่าง ๆ ในกระบวนการพัฒนาซอฟต์แวร์



รูปที่ 3.2 ตัวอย่างสคริปต์/คำสั่งของ Robot Framework

โดย Robot Framework มีส่วนหลักในไฟล์สคริปต์ของ Robot Framework (.robot) ซึ่งช่วยในการกำหนดค่า และสร้างโครงสร้างการทดสอบ ดังนี้

1) ***** Settings ***** เป็นส่วนที่ใช้ในการเรียกใช้งานไลบรารีต่าง ๆ เพื่อให้สามารถใช้คำสั่งที่เกี่ยวข้องกับระบบปฏิบัติการได้ โดยใช้สัจเวิร์ด และค่าที่ต้องการกำหนดให้กับการรันหรือการทำงานโดยรวมของ Robot Framework

```
1  *** Setting ***
2  Library    SeleniumLibrary
3
```

รูปที่ 3.3 ตัวอย่างการกำหนด Setting

จากรูปตัวอย่างข้างต้น

- “Library” เป็นการกำหนดไลบรารีที่ต้องการใช้ในการรันหรือการทดสอบ เพื่อเรียกใช้ Keywords จากไลบรารีเพื่อทำงานต่าง ๆ จากตัวอย่างใช้สัจเวิร์ด SeleniumLibrary (ในส่วนนี้จะอธิบายในหัวข้อ 3.2.2)

2) ***** Variables ***** เป็นการกำหนดค่าตัวแปรที่ใช้ในการเก็บข้อมูลหรือค่าที่ใช้ในการทำงานของ Test Cases หรือ Keywords ต่าง ๆ และช่วยให้สามารถเปลี่ยนค่าของตัวแปรได้ง่ายเมื่อต้องการปรับปรุงแก้ไขค่าในทีเดียวแทนที่จะต้องแก้ไขทุกส่วนที่ใช้ค่านั้น

```
4  *** Variables ***
5  ${BROWSER}    chrome
6  ${URL}        https://google.com
```

รูปที่ 3.4 ตัวอย่างการกำหนด Variables (ตัวแปร)

```

13  *** Test Cases ***
14  chrome_test
15      #เปิดเบราว์เซอร์
16      Open Browser    ${URL}    ${BROWSER}
17      Sleep    1s
18      #ขยายหน้าต่าง
19      Maximize Browser Window
20      Sleep    3s
21      #ปิดเบราว์เซอร์
22      Close Browser

```

รูปที่ 3.5 ตัวอย่างการใช้งาน Variables (ตัวแปร) ใน Test Case

จากรูปตัวอย่างข้างต้น

- \${BROWSER} และ \${URL} คือตัวแปรที่กำหนดในส่วนของ Variables (ตัวแปร)
- \${BROWSER} และ \${URL} ใช้ในการเก็บค่า เบราว์เซอร์ และ URL
- ใน Test Cases “chrome_test” สามารถเรียกใช้ค่าตัวแปรนี้ในการใช้งานได้

3) *** Keywords *** คือชุดคำสั่งหรือการกระทำที่กำหนดขึ้นเพื่อใช้ในการสร้างเทสเคส หรือการจัดการกับคำสั่งในเทสเคส คีย์เวิร์ดหรือคำสั่งเหล่านี้เป็นที่กำหนดโดยตัวผู้ใช้เอง และสามารถรวมกันเป็นชุดของคำสั่งที่มีวัตถุประสงค์ในการทำงานเฉพาะในเนื้อหาของ Test Cases ที่ผู้ใช้เขียน

```

8  *** Keywords ***
9  Open Chrome
10     Open Browser    https://google.com    chrome
11     Sleep    1s

```

รูปที่ 3.6 ตัวอย่างการใช้งาน Keywords

จากรูปตัวอย่างข้างต้น

- Open Chrome เป็นชื่อของ Keywords ที่ผู้ใช้กำหนดเอง
 - Open Browser และ Sleep เป็นคีย์เวิร์ด และคำสั่งที่ถูกใช้เพื่อดำเนินการต่าง ๆ
- ในการทำงานที่ Keyword กำหนด

4) *** Test Cases *** เป็นส่วนหนึ่งในไฟล์ที่สร้างขึ้นเพื่อเขียน และกำหนดเทสเคส (Test Cases) ที่ต้องการทดสอบในโปรแกรมหรือระบบที่กำลังพัฒนา โดยส่วนนี้จะเป็นที่ที่ระบุว่าการทดสอบอะไร และวิธีการทดสอบเป็นอย่างไร โดยใช้คีย์เวิร์ด และคำสั่งที่เป็นไปตามโครงสร้างของ Robot Framework โดยโครงสร้างของ Test Cases จะประกอบด้วยส่วนหลัก ๆ ดังนี้

- Test Case Name เป็นชื่อของเทสเคส เพื่อระบุว่าการทดสอบนี้ทำงานอะไร ชื่อนี้จะถูกใช้ในการระบุเทสเคสเมื่อรันการทดสอบ

- Documentation (หรือ การ Comment) ส่วนนี้สามารถใช้ในการเพิ่มคำอธิบายหรือคำอธิบายเพิ่มเติมเกี่ยวกับเทสเคส เช่น วัตถุประสงค์ของการทดสอบ ข้อมูลทดสอบที่ใช้ คาดหวังว่าจะเกิดอะไรขึ้น เป็นต้น

- Test Steps (Keywords) เป็นส่วนที่ระบุคีย์เวิร์ด และคำสั่งที่ใช้ในการทดสอบ เช่น การคลิกที่ปุ่ม กรอกข้อมูล การชะลอเวลาการดำเนินการถัดไป เป็นต้น

ดังนั้น หากมอง Test Cases ในโครงสร้างเบื้องต้น จะมีลักษณะดังนี้

```

13  *** Test Cases ***
14  chrome_test
15      #เปิดเบราว์เซอร์
16      Open Browser    ${URL}    ${BROWSER}
17      Sleep    1s
18      #ขยายหน้าต่าง
19      Maximize Browser Window
20      Sleep    3s
21      #ปิดเบราว์เซอร์
22      Close Browser
23
24  chrome_test1
25      #เปิดเบราว์เซอร์
26      Open Chrome
27      Sleep    1s
28      #ขยายหน้าต่าง
29      Maximize Browser Window
30      Sleep    3s
31      #ปิดเบราว์เซอร์
32      Close Browser

```

รูปที่ 3.7 ตัวอย่างโครงสร้างของ Test Case

จากรูปตัวอย่างข้างต้น

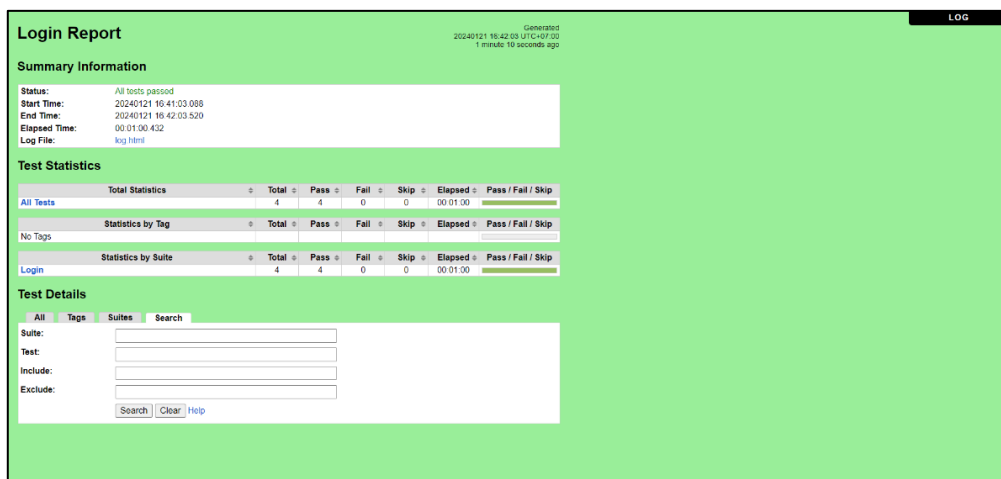
- chrome_test และ chrome_test1 เป็นชื่อของเทสเคส
- Documentation (หรือ การ Comment) ใช้ในการเพิ่มคำอธิบายหรือคำอธิบายเพิ่มเติมเกี่ยวกับเทสเคส โดยใช้ เครื่องหมาย “#”

- “Open Browser”, “Sleep”, “Maximize Browser Window” และ “Close Browser” เป็นคีย์เวิร์ด และคำสั่งที่ใช้ในการทดสอบ

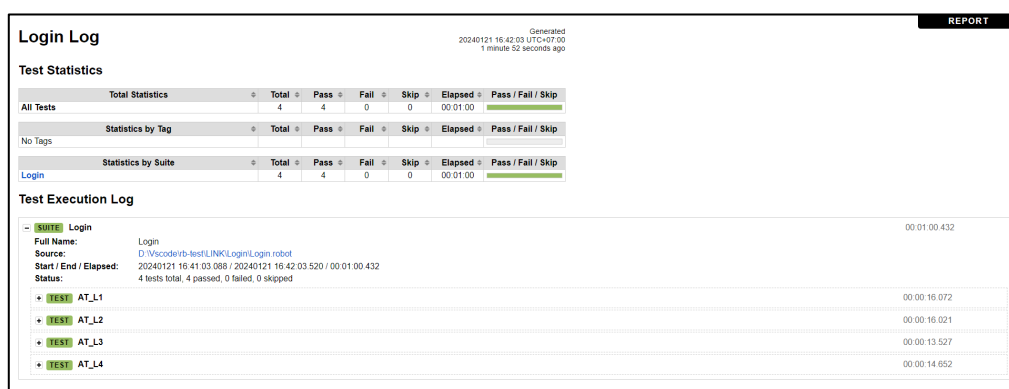
หลังจากรัน Test Case ใน Robot Framework เสร็จเรียบร้อยแล้ว จะได้รับไฟล์ Report และไฟล์ Log ซึ่งมักนำไปใช้งานด้วยวิธี ดังนี้

- ไฟล์ Report (Report File) เป็นผลลัพธ์ที่แสดงผลของการรัน Test Case ทั้งหมด รายงานนี้จะแสดงผลสรุปเกี่ยวกับผลการทดสอบที่ปรากฏ เช่น จำนวน Test Case ที่ผ่าน และไม่ผ่าน ข้อผิดพลาด (if any) และข้อมูลอื่น ๆ

- ไฟล์ Log (Log File) เป็นไฟล์ที่บันทึกข้อมูลการทดสอบเพิ่มเติม รวมถึงข้อมูลการทดสอบแต่ละขั้นตอนใน Test Case รวมถึงข้อมูลเช่น timestamps, actions ที่ทำโดย Robot Framework และข้อผิดพลาด (if any)



รูปที่ 3.8 ตัวอย่างไฟล์ Report จากการทดสอบ



รูปที่ 3.9 ตัวอย่างไฟล์ Log จากการทดสอบ

การติดตั้ง Robot Framework สามารถทำได้ด้วยขั้นตอนต่อไปนี้

1) ตรวจสอบการติดตั้ง Python Robot Framework ใช้ Python เป็นภาษาหลักในการทำงาน ดังนั้นควรตรวจสอบว่ามี Python ติดตั้งอยู่แล้วหรือไม่ โดยใช้คำสั่ง `python --version` หรือ `python3 --version` ใน Command Line (หรือ Terminal บน macOS หรือ Linux) เพื่อตรวจสอบเวอร์ชัน Python ที่มีในเครื่อง

2) ติดตั้ง pip (Package Installer for Python) หากยังไม่มี pip ให้ทำการติดตั้งก่อน โดยปกติแล้ว pip จะถูกติดตั้งพร้อมกับ Python สำหรับเวอร์ชัน Python 2.x แต่ สำหรับ Python 3.x อาจต้องใช้ pip3 แทน

3) ติดตั้ง Robot Framework ใช้คำสั่งต่อไปนี้ใน Command Line หรือ Terminal เพื่อติดตั้ง Robot Framework และเครื่องมือที่เกี่ยวข้องผ่าน pip

```
pip install robotframework
```

4) ติดตั้งเครื่องมือสำหรับการเขียน Test Cases และการรัน Robot Framework มีเครื่องมือหลายตัวที่ช่วยในการเขียน และรัน Test Cases ได้ง่ายขึ้น สามารถติดตั้งเครื่องมือเหล่านี้ได้ ดังนี้

- RIDE (Robot Framework IDE) เป็นโปรแกรมที่ให้ GUI สำหรับการเขียน และรัน Test Cases โดยไม่ต้องเขียนสคริปต์ด้วยตนเอง

```
pip install robotframework-ride
```

- Robot Framework Command Line Interface (CLI) สำหรับการรัน Test Cases ผ่าน Command Line โดยใช้คำสั่ง `robot` (ในส่วนของการทำงานโปรเจกต์นี้ ใช้ CLI ในการรัน Test Cases)

5) ตรวจสอบการติดตั้ง หลังจากติดตั้งเสร็จสิ้น สามารถตรวจสอบว่า Robot Framework ได้ถูกติดตั้งอย่างถูกต้องด้วยการใช้คำสั่ง

```
robot --version
```

6) เริ่มใช้งาน สามารถเริ่มเขียน และรัน Test Cases โดยใช้เครื่องมือต่าง ๆ ที่ติดตั้งไว้ หากใช้ RIDE สามารถเปิดโปรแกรมนั้นขึ้นมาแล้วเริ่มเขียน Test Cases ได้ทันที

3.2.2 SeleniumLibrary

SeleniumLibrary เป็นไลบรารี (Library) สำหรับ Robot Framework ที่ใช้ในการทำอัตโนมัติ และการทดสอบเว็บแอปพลิเคชันผ่านเบราว์เซอร์ เป็นเครื่องมือที่มีความสามารถในการควบคุม และจำลองการกระทำของผู้ใช้งานบนเว็บเบราว์เซอร์ เพื่อใช้ในการทดสอบความสมบูรณ์ และความถูกต้องของเว็บแอปพลิเคชัน

โดย SeleniumLibrary มีคุณสมบัติที่สามารถทำสิ่งต่าง ๆ ได้ เช่น

- การเปิด และปิดเบราว์เซอร์ สามารถเปิด และปิดเบราว์เซอร์ที่รองรับได้ เช่น Chrome, Firefox, Edge, Safari เป็นต้น
- การควบคุมการทำงานบนเว็บ สามารถควบคุมการกระทำบนเว็บได้ เช่น คลิกที่ลิงก์ กรอกข้อมูลในฟอร์ม ตรวจสอบเนื้อหาเว็บ เป็นต้น
- การจัดการกับไฟล์ และดาวน์โหลด สามารถอัปโหลด และดาวน์โหลดไฟล์ได้ผ่านการใช้ SeleniumLibrary

การใช้งาน SeleniumLibrary ใน Robot Framework มีขั้นตอนหลัก ๆ ดังนี้

- 1) การเรียกใช้ SeleniumLibrary ตรวจสอบการติดตั้ง SeleniumLibrary ด้วยคำสั่ง

```
pip install robotframework-seleniumlibrary
```

- 2) หลังจากนั้นใน Test Case ที่ต้องการใช้งาน SeleniumLibrary ให้เพิ่มการเรียกใช้ไลบรารีในส่วนของ Settings ด้วยคำสั่ง

```
*** Settings ***
Library      SeleniumLibrary
```

- 3) การเรียกใช้ WebDriver เลือก WebDriver ที่ต้องการใช้งาน เช่น ChromeDriver สำหรับ Google Chrome (ในส่วนของการทำงานโปรเจกต์นี้ ใช้ ChromeDriver และจะอธิบายใน

หัวข้อ 3.2.3) GeckoDriver สำหรับ Mozilla Firefox และอื่น ๆ ที่สามารถกำหนดให้ WebDriver ทำงานกับเบราว์เซอร์ตามที่ต้องการ สามารถทำได้โดยใช้คำสั่ง Open Browser

```
Open Browser    https://www.example.com    chrome
```

4) การใช้ Keywords ใน SeleniumLibrary SeleniumLibrary มี Keywords มากมายให้ใช้งาน เช่น Click Element, Input Text, Select From List by Value, Wait Until Element is Visible, เป็นต้น ซึ่งสามารถใช้ Keywords เหล่านี้ใน Test Steps เพื่อทำการตรวจสอบหรือจำลองการทำงานบนเว็บเบราว์เซอร์

```
Click Element    //button[@id='submit-button']
Input Text       //input[@name='username']    MyUsername
Select From List by Value    //select[@name='country']    US
Wait Until Element is Visible    //div[@class='loading-spinner']
```

5) การปิด WebDriver เมื่อเสร็จสิ้นการทดสอบหรือใช้งาน WebDriver เสร็จแล้วควรทำการปิด WebDriver ด้วยคำสั่ง Close Browser

```
Close Browser
```

6) การสร้าง Test Cases สามารถเรียงเรียง Keywords ที่ใช้งานกับ SeleniumLibrary เพื่อสร้าง Test Cases ตามที่ต้องการทดสอบบนเว็บ

```
*** Test Cases ***
```

```
Test Login
```

```
Open Browser    https://www.example.com    chrome
Input Text      //input[@name='username']    myusername
Input Text      //input[@name='password']    mypassword
Click Element    //button[@id='login-button']
Wait Until Page Contains    Welcome, User!
```

- การรัน Test Cases เมื่อเขียน Test Cases เสร็จแล้ว สามารถใช้คำสั่ง robot บน Command Line หรือ Terminal เพื่อรัน Test Cases ได้

```
robot testfile.robot
```

3.2.3 Chrome Driver

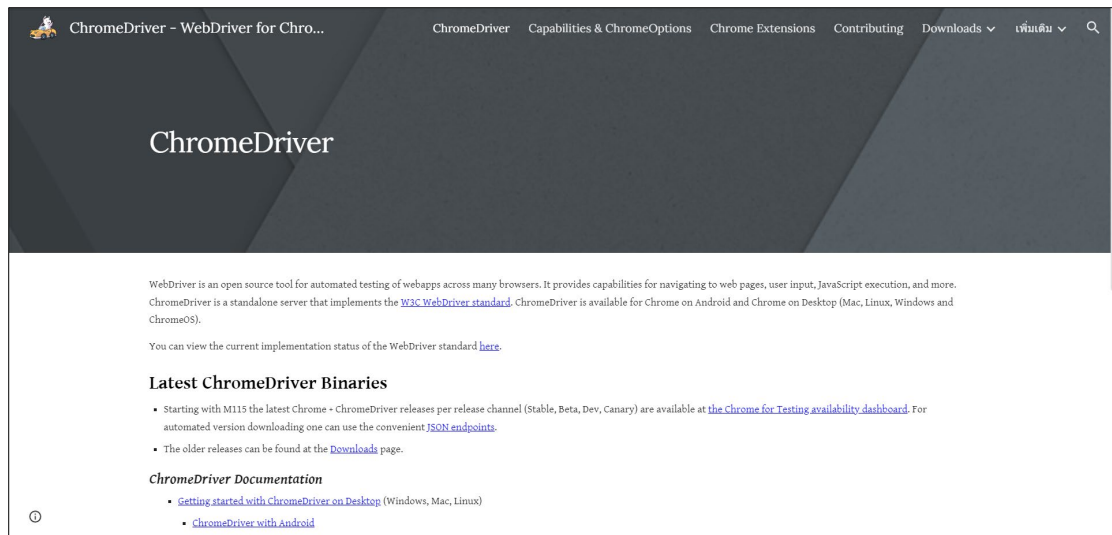
Chrome Driver เป็นเครื่องมือหรือโปรแกรมที่เป็นส่วนหนึ่งของ Selenium WebDriver ซึ่งถูกพัฒนาขึ้นเพื่อใช้ในการควบคุม และทดสอบเบราว์เซอร์ Google Chrome อย่างอัตโนมัติ และทำการออโตเมชันบนเว็บได้

ChromeDriver ใช้เพื่อเชื่อมต่อ และควบคุมการทำงานของเบราว์เซอร์ Google Chrome ผ่านอินเทอร์เฟซของ Selenium WebDriver ที่ใช้สำหรับการทดสอบ และออโตเมชันเว็บ สามารถใช้งาน ChromeDriver ได้ ดังนี้

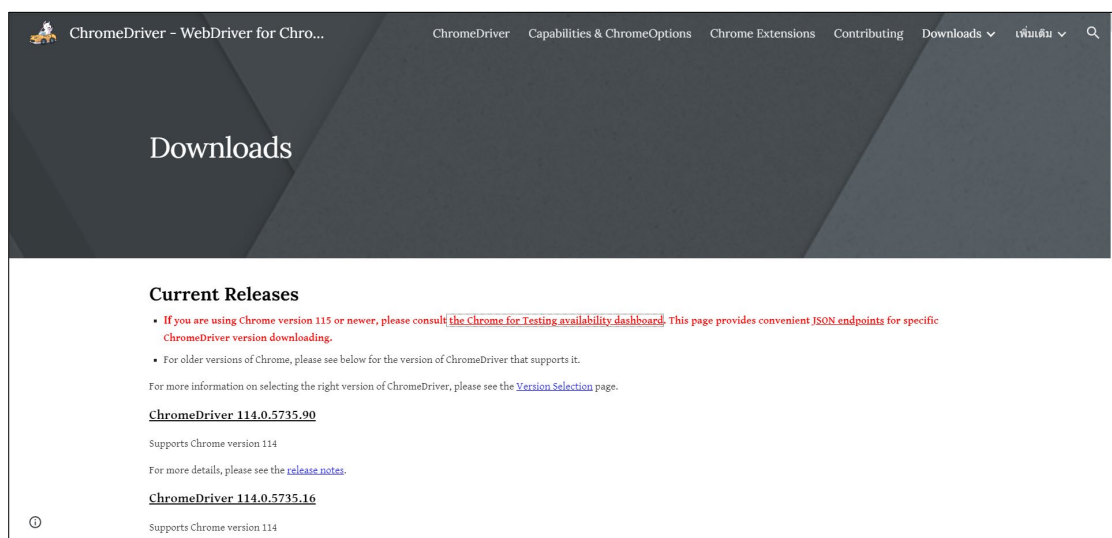
- ChromeDriver ช่วยในการเปิดหน้าต่างเบราว์เซอร์ Google Chrome ให้กับการทดสอบหรือออโตเมชัน
- สามารถใช้ ChromeDriver เพื่อควบคุมการกระทำต่าง ๆ บนเบราว์เซอร์ เช่น คลิกที่ลิงค์ กรอกข้อมูลในฟอร์ม กดปุ่ม เปลี่ยนหน้าเว็บ เป็นต้น
- ChromeDriver ช่วยในการตรวจสอบสถานะ และข้อมูลในเว็บได้อย่างอัตโนมัติ เช่น การตรวจสอบข้อความ การตรวจสอบค่าในฟิลด์ การตรวจสอบข้อมูลในตาราง ฯลฯ
- เมื่อทำการทดสอบหรือออโตเมชันเสร็จสิ้น สามารถใช้ ChromeDriver เพื่อปิดหน้าต่างเบราว์เซอร์ได้

ในส่วนของการติดตั้ง ChromeDriver เป็นขั้นตอนสำคัญเมื่อต้องการใช้ Selenium WebDriver เพื่อควบคุมเบราว์เซอร์ Chrome จากภาษาโปรแกรมที่ต้องการ เช่น Python, Java, หรือภาษาอื่น ๆ

1) ดาวน์โหลด ChromeDriver ไปที่เว็บไซต์ดาวน์โหลดของ ChromeDriver ที่ <https://sites.google.com/chromium.org/driver/> และเลือกเวอร์ชันของ ChromeDriver ที่ตรงกับเวอร์ชันของ Google Chrome ที่คุณมีในเครื่อง

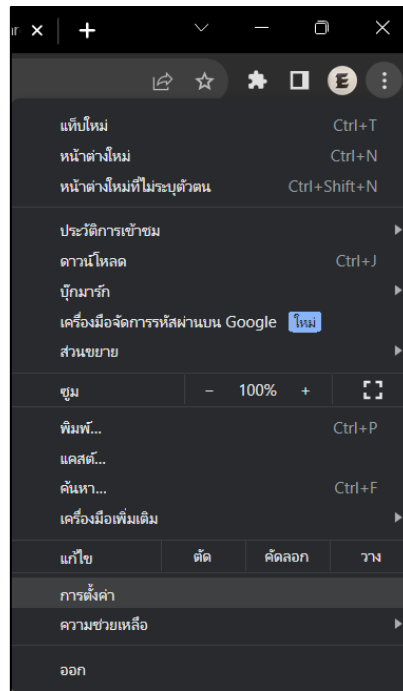


รูปที่ 3.10 หน้าเว็บของ ChromeDriver

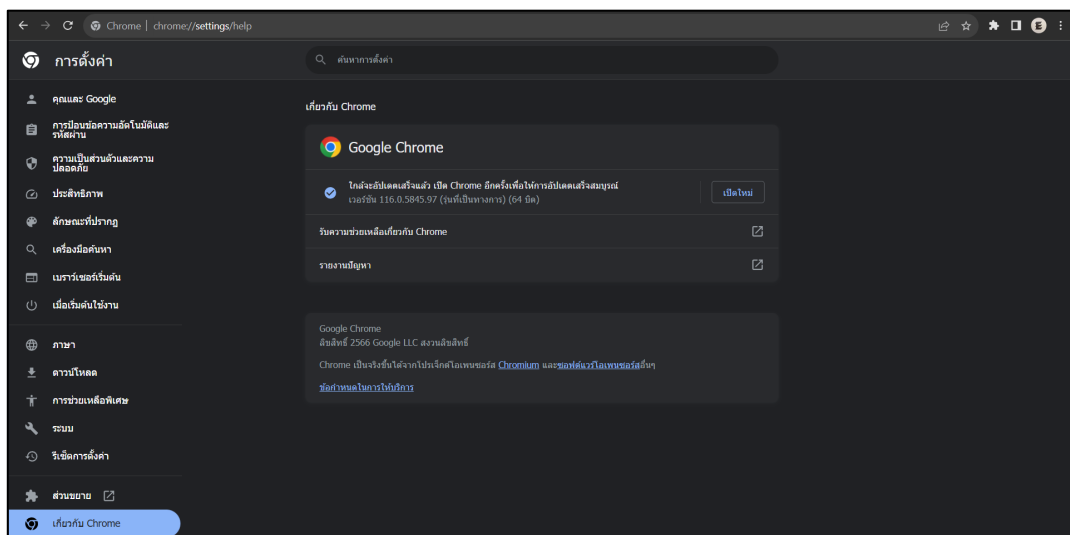


รูปที่ 3.11 หน้า Downloads

2) สามารถตรวจสอบเวอร์ชันของ Chrome ที่มีในเครื่องโดยการ เข้าไปที่ “ตั้งค่า (Settings)” ของ Chrome -> เกี่ยวกับ Chrome (About Chrome)




รูปที่ 3.12 “ตั้งค่า (Settings)” ของ Chrome



รูปที่ 3.13 ตรวจสอบเวอร์ชันของ Chrome ที่มีในเครื่อง

3) ในกรณีที่เวอร์ชันของ Chrome ที่มีในเครื่อง มีเวอร์ชันที่สูงกว่า หรือใหม่กว่าในหน้าเว็บ “Downloads” ให้เข้าไปที่ลิงค์ <https://googlechromelabs.github.io/chrome-for-testing/> และเลือกเวอร์ชันที่ตรงกับ Chrome ที่มีในเครื่อง

Chrome for Testing availability 

This page lists the latest available cross-platform Chrome for Testing versions and assets per Chrome release channel.

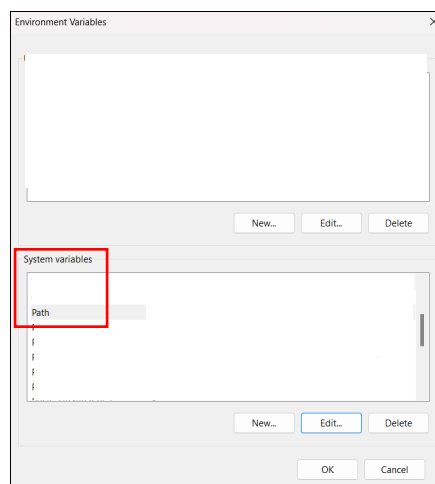
Consult [our JSON API endpoints](#) if you're looking to build automated scripts based on Chrome for Testing release data.

Last updated @ 2023-08-23T04:08:45.846Z

Channel	Version	Revision	Status
Stable	116.0.5845.96	r1160321	✓
Stable (upcoming)	116.0.5845.98	r1160321	✗
Beta	117.0.5938.11	r1181205	✓
Dev	118.0.5951.0	r1183763	✓
Canary	118.0.5964.0	r1186366	✓

รูปที่ 3.14 หน้าเว็บในการดาวน์โหลด ChromeDriver เวอร์ชันล่าสุด

4) เพิ่ม ChromeDriver เข้า PATH (ถ้าต้องการ) เพื่อให้สามารถเรียกใช้ ChromeDriver จากทุกที่ในระบบ อาจจะต้องทำการเพิ่มเส้นทางที่แสดงถึงไฟล์ ChromeDriver เข้าไปใน Environment Variables (PATH) ของระบบปฏิบัติการ



รูปที่ 3.15 เพิ่ม path ที่ชื่อว่า “Path” ในส่วนของ “System variables”

C:\Python\Python310\Scripts

รูปที่ 3.16 Path ที่เรียกใช้ ChromeDriver (ในที่นี้ขึ้นอยู่กับผู้ใช้ในการกำหนด Path ที่ต้องการ)

3.2.4 Microsoft Excel

Microsoft Excel พัฒนาโดยบริษัท Microsoft มีการใช้งานหลากหลายในการบันทึกข้อมูลการทดสอบทั้งในลักษณะการทดสอบด้วยมือ และระบบอัตโนมัติ รวมถึงการบันทึกผลลัพธ์ของกระบวนการทดสอบ การจัดการ และวิเคราะห์ข้อมูลที่เกี่ยวข้องกับการทดสอบ และตรวจสอบคุณภาพของซอฟต์แวร์ ได้แก่

- การวางแผน และการจัดการกระบวนการทดสอบ Excel สามารถใช้ในการสร้างแผนการทดสอบ กำหนดลำดับการทดสอบ รวมไปถึงการบันทึก และติดตามความคืบหน้าของกระบวนการทดสอบ

- การจัดการข้อมูลทดสอบ ช่วยในการจัดเก็บ และจัดการข้อมูลทดสอบ เช่น การสร้างชุดข้อมูลทดสอบ การบันทึกผลการทดสอบ และการจัดรูปแบบข้อมูลเพื่อให้ง่ายต่อการวิเคราะห์

- การจัดทำรายงานการทดสอบ ช่วยในการจัดทำรายงานผลการทดสอบ เพื่อแสดงข้อมูล และผลลัพธ์ของการทดสอบให้กับทีมพัฒนาหรือผู้เกี่ยวข้องอื่น ๆ

- การจัดการข้อมูลการทดสอบอัตโนมัติ (Automated Testing) เมื่อมีการทดสอบอัตโนมัติ ผลลัพธ์จะถูกบันทึกในรูปแบบของข้อมูล ซึ่งอาจใช้ Excel เพื่อจัดเก็บ และวิเคราะห์ผลลัพธ์การทดสอบอัตโนมัติ

- การจัดการกับข้อมูลคุณภาพ ใช้เพื่อจัดเก็บข้อมูลเกี่ยวกับคุณภาพของซอฟต์แวร์ เช่น รายชื่อบั๊ก รายงานความสำเร็จของการทดสอบ เป็นต้น

โดยมีการนำตัวอย่างเทสเคสที่มีในการใช้ในการทดสอบมาเป็นตัวอย่าง ในโปรแกรมไมโครซอฟต์เอกเซล (Microsoft Excel) ในรูปแบบของตาราง เพื่อให้ง่ายต่อการบันทึก และติดตามความคืบหน้าของกระบวนการทดสอบ

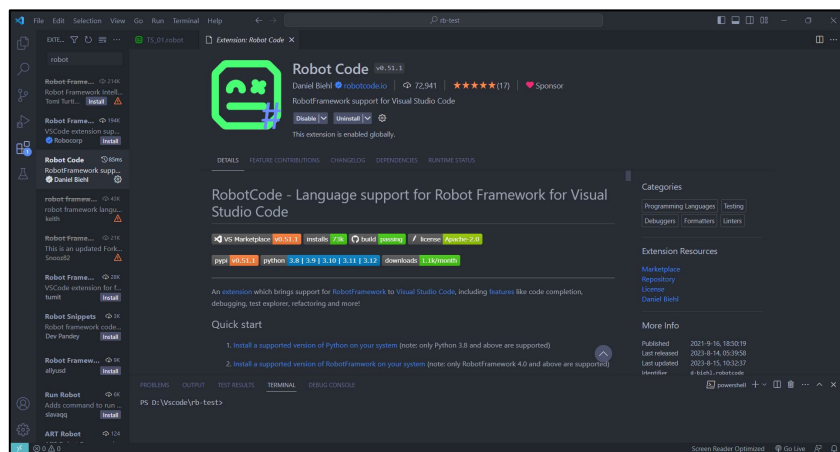
Test Case ID	Test Case Description	Test Data / Input	Test Step	Expected Results	Actual Results	Status/Note
U_1.1	ผู้ใช้งาน Login Username: admin@idk.com Password: 123456789	1. 1551 'Username' Like 'Password' X10 Test Data Username: admin@idk.com Password: 123456789	1. 1551 'Username' Like 'Password' X10 Test Data 2. คลิกปุ่ม 'Login'	ระบบเข้าสู่ระบบ Home ดี Username: admin@idk.com Password: 123456789	ระบบเข้าสู่ระบบ Home ดี Username: admin@idk.com Password: 123456789	Pass
U_1.2	ผู้ใช้งาน Login Username: 123 Password: admin	1. 1551 'Username' Like 'Password' X10 Test Data Username: 123 Password: admin	1. 1551 'Username' Like 'Password' X10 Test Data 2. คลิกปุ่ม 'Login'	ระบบเข้าสู่ระบบ Home ดี Username: 123 Password: admin	ระบบเข้าสู่ระบบ Home ดี Username: 123 Password: admin	Pass
U_1.3	ผู้ใช้งาน Login Username: Password	1. 'Username' Like 'Password' 104511 Username: Password	1. 'Username' Like 'Password' 104511 2. คลิกปุ่ม 'Login'	ระบบเข้าสู่ระบบ Home ดี Username: Password Password: Password	ระบบเข้าสู่ระบบ Home ดี Username: Password Password: Password	Pass

รูปที่ 3.17 หน้าโปรแกรม Microsoft Excel

ทั้งนี้ Test Case ไม่ได้มีรูปแบบที่แน่นอน เนื่องจากสามารถเขียน และออกแบบได้ในหลายลักษณะขึ้นอยู่กับประเภทของซอฟต์แวร์ และวัตถุประสงค์ของการทดสอบ

3.2.5 Extension Robot Code

ต้องติดตั้งส่วนขยาย (extension) ของ Robot Framework ใน Visual Studio Code (VSCode) ก่อนเพื่อให้สามารถเขียน และจัดการกับไฟล์สคริปต์ของ Robot Framework ได้

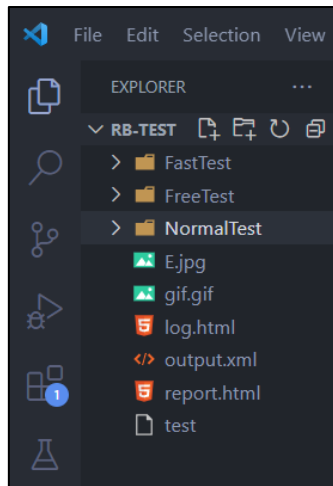


รูปที่ 3.18 ติดตั้ง Extensions Robot Code

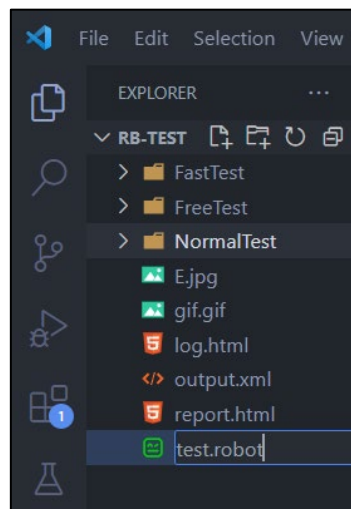
ในรูปตัวอย่างนี้ คือ Extensions ที่ชื่อว่า “Robot Code” เป็นส่วนขยายที่ใช้สำหรับรองรับการเขียนสคริปต์ Robot Framework ใน Visual Studio Code (VSCode)

การสร้างไฟล์สคริปต์ Robot Framework ใน Visual Studio Code (VSCode) สามารถทำได้ ดังนี้

- 1) เปลี่ยนนามสกุลไฟล์ หลังจากที่เราสร้างไฟล์ใหม่ ให้เปลี่ยนนามสกุลไฟล์ให้เป็น “.robot”

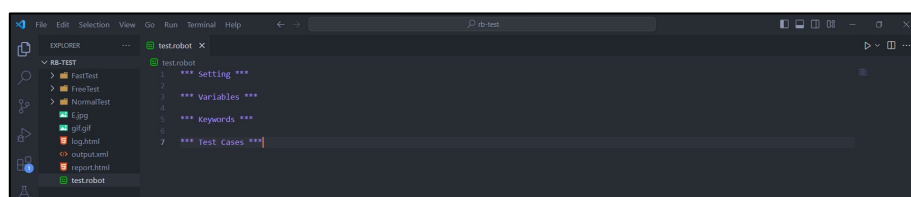


รูปที่ 3.19 ไฟล์ที่สร้างขึ้น จากรูปคือไฟล์ “test”



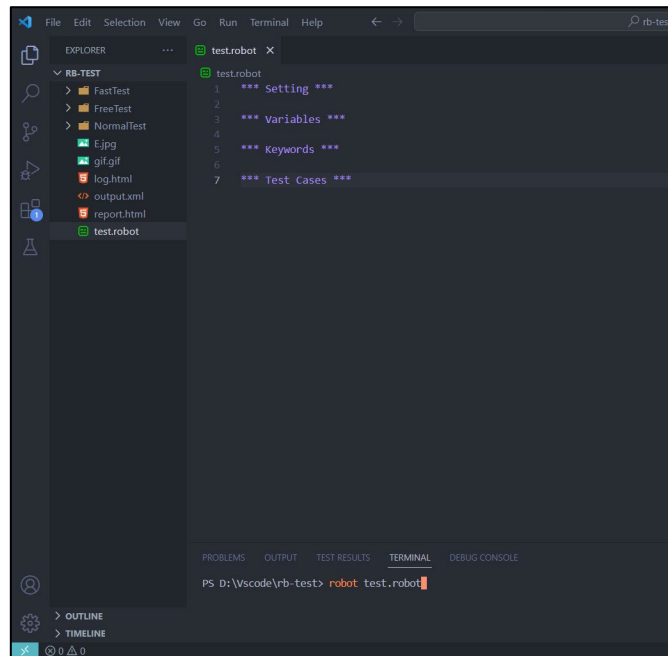
รูปที่ 3.20 เติมนามสกุล “.robot” ให้กับไฟล์ “test”

2) เริ่มเขียนสคริปต์ สามารถเริ่มเขียนสคริปต์ Robot Framework ในไฟล์นี้ได้ โดย เพิ่มส่วน *** Settings *** , *** Variables *** , *** Keywords *** , และ *** Test Cases *** ตามความต้องการของในทดสอบ (ในส่วนต่าง ๆ เหล่านี้ จะนำไปอธิบายในหัวข้อของ Robot Framework)



รูปที่ 3.21 เพิ่มส่วน *** Settings *** , *** Variables *** , *** Keywords *** , และ *** Test Cases ***

3) การรัน และทดสอบ หากต้องการรัน และทดสอบไฟล์สคริปต์ Robot Framework ที่สร้าง เปิดเทอร์มินัลหรือหน้าต่าง Terminal ใน Visual Studio Code (VSCode) และใช้คำสั่ง robot เพื่อรันไฟล์สคริปต์ ตัวอย่างเช่น robot test.robot



รูปที่ 3.22 เปิด Terminal และใช้คำสั่ง robot เพื่อรันไฟล์สคริปต์